Numerical Reproducibility and Stability of Large-Scale Simulations on Exascale Architectures

Michela Taufer, University of Delaware

1. Exascale challenges

There is no doubt that exascale architectures will be multi-core platforms, with the number of cores expected to increase substantially. The increased number of cores brings a radical change to what scientists can study on exascale systems; it also exposes and emphasizes a well-known and so far often ignored problem: large-scale, massively parallel simulations lacking numerical reproducibility and stability. In other words, the numerical reproducibility and stability of simulation results cannot be assured on large multi-core platforms because computers limit the representation of floating-point arithmetic; threads are scheduled randomly among cores; and scientists fail to consider mathematical properties in their applications (e.g., the monotonicity of rounding errors) [1,2]. In the past, both hardware and runtime software solutions have been proposed to the point that the scientific community has considered this problem solved. We claim that existing solutions will not work for exascale systems, and therefore, there is the need for new runtime software solutions that can transform the way large-scale simulations are performed on multi-core platforms by leveraging numerical reproducibility and stability.

As exascale architectures become reality, we will assist in the increasing number of applications simulated on them. Application domains impacted by these architectures include molecular dynamics [3-5], quantum chemistry [6], fluid dynamics [7], and Monte Carlo methods [8,9]. Ignoring the numerical reproducibility and stability problem today could result in making erroneous conclusions about the final states of large-scale simulations tomorrow. The introduction of GPU-based platforms has already revealed the problem with numerical reproducibility and stability because of the large number of cores working in concert on a single simulation; this problem will become more obvious as the number of cores in CPU-based systems increases. Some developers have confused this problem with a simple lack of precision and have proposed hardware and software solutions to improve the precision. Vendors have proposed hardware solutions, such as including implementations of 64-bit floating-point arithmetic in GPUs, while scientists have tackled the problem by manually identifying and integrating single and double precisions in separate code parts. However, numerical reproducibility and stability cannot be guaranteed just by increasing precision, as shown in previous work on GPUs [10,11] and general multi-core CPU-based architectures [12]. The numerical reproducibility and stability problem in large-scale simulations on multi-core platforms cannot be solved simply with the knowledge and tools currently available; instead, new, transformative methods are required.

2. Proposed approach, its uniqueness, and maturity

Two specific research directions are a matter of great priority before the exascale era becomes reality. At the runtime software level, there is the need to leverage numerical reproducibility and stability in simulations on multi-core platforms by (a) creating modeling patterns and mathematical methods to capture and prevent propagating rounding errors that can cause numerical irreproducibility and instability in simulations; (b) creating automatic techniques that identify the code segments that are significantly impacting the reproducibility and stability of simulation results; and (c) integrating modeling patterns, mathematical methods, and automatic techniques into an automatic runtime software solution that intelligently and efficiently uses the two in calculations. At the application level, there is the need to use the new runtime software solutions to improve numerical reproducibility and stability of results in relevant large-scale simulations of physical systems that involve the physical system state to evolve algorithmically over time (so-called chaotic applications).

The major flaw in traditional floating-point numbers is that an accurate representation of values with many significant bits is not possible, as the less significant bits may be truncated. This results in irreproducibility, and the instability of large-scale simulations has been addressed in the past by increasing the hardware precision (e.g., from 32-bit to 64-bit representations). Computationally, increasing the precision without considering important properties such as the monotonicity of rounding errors, or without maintaining correct relationships amongst precisions across the simulation components, can ultimately result in poorer, rather than increased, accuracy. Therefore, there is the need to go beyond rigid hardware solutions by studying the impact of error modeling patterns on reproducibility and stability. There is also the need to go beyond manual software solutions by defining the automatic techniques that are able to identify the code components principally causing instability and irreproducibility in simulations on multi-core platforms. We propose to use what we defined as *composite precision* to handle floating-point values as "clusters" of contiguous significant bits with a large number of zeros separating them [10]. This approach enables a more accurate representation of values with separate single precision floating-point numbers - intuitively one for

each cluster, although not necessarily - for which each cluster of bits corresponds to a portion of the mantissa. The algorithms used for performing composite precision floating-point operations are defined in terms of multiple single precision operations. The algorithms deploying our runtime software solutions should be "self-compensating"; they should perform the calculation as well as keep track of inherent rounding errors. There is the common acknowledgment that not all the parts of the code contribute to rounding errors in simulations in the same way. Therefore, there is the need for studying automatic techniques to detect, isolate, and improve those parts that have a higher impact on numerical reproducibility and stability. The generation of good techniques diagnosing loss in result accuracy must be automatic and must be sensitive to the type of data processed. Runtime software solutions can automate both result diagnosis and code synthesis, controlling the time to improve numerical reproducibility and stability.

3. Applicability and effort

Trends in computer architecture make a compelling case for the development of our proposed approach to automatically and efficiently address numerical reproducibility and stability beyond petascale architecture into exascale. In absence of exascale platforms, and because the numerical reproducibility and stability problem is already emphasized by multi-threaded executions, these types of executions can be the starting point for modeling patterns and mathematical methods capable of increasing the result accuracy for these executions, without having to redesign hardware components. There are very diverse scientific domains that can potentially benefit from the proposed approach. We target a broader range of chaotic applications, such as the simulation of global seismic wave propagations, the simulation of solar systems in real-time 3D, and fluid dynamics simulations. In particular, emphasis is initially put on deterministic chaotic applications to address the need to identify when and how the behavior of chaotic systems diverges due to rounding error propagations, rather than the intrinsic system properties [13]. Our research builds on our long experience in scientific simulations and algorithms, multi-core platforms and multi-threaded programming, and interdisciplinary work; thus, we are uniquely qualified to participate in this effort to assure the numerical reproducibility and stability of large-scale simulations on multi-core exascale platforms.

4. References

[1] D.H. Bailey. High-precision floating-point arithmetic in scientific computing. *IEEE Computing in Science and Engineering.* 54–61, 2005.

[2] N.J. Higham. Accuracy and Stability of Numerical Algorithms. *Society for Industrial and Applied Mathematics*. Philadelphia, PA, USA, Second Edition, 2002. ISBN 0-89871-521-0.

[3] M.S. Friedrichs, P. Eastman, V. Vaidyanathan, M. Houston, S. LeGrand, A.L. Beberg, D.L. Ensign, C.M. Bruns, and V.S. Pande. Accelerating molecular dynamic simulation on graphics processor units. *J. Comput. Chem.* **30**: 864 – 872, 2009.

[4] J. Stone, J. Phillips, P. Freddolino, D. Hardy, L. Trabuco, and K. Schulten. Accelerating molecular modeling applications with graphics processors. *J. Comput. Chem.* **28**: 2618 – 2640, 2007.

[5] D. Xu, M.J. Williamson, and R.C, Walker. Advancements in molecular dynamics simulations of biomolecules on graphical processing units. *Ann. Rep. Comp. Chem.* **6**: 2 – 19, 2010.

[6] A.W. Goetz, T. Woelfe, and R.C. Walker. Quantum chemistry on graphics processing units. *Ann. Rep. Comp. Chem.* **6**: 21 – 35, 2010.

[7] A. Corrigan, F. Camelli, R. Loehner, and J. Wallin: Running Unstructured Grid CFD Solvers on Modern Graphics Hardware. In *Proceedings of the 19th AIAA Computational Fluid Dynamics Conference*. 2009.

[8] A. Lee, C. Yau, M. Giles, A. Doucet, and C. Holmes. On the utility of graphics cards to perform massively parallel simulation with advanced Monte Carlo methods. *J. Comp. & Graphical Statistics.* **19**(4), 2010.

[9] T. Preis, P. Virnau, W. Paul, and J.J. Schneider. GPU accelerated Monte Carlo simulation of the 2D and 3D Ising model. *J. Comp. Phys.* **228**(12): 4468 – 4477, 2009.

[10] M. Taufer, O. Padron, P. Saponaro, and S. Patel: Improving Numerical Reproducibility and Stability in Large-Scale Numerical Simulations on GPUs. In *Proceedings of the IEEE/ACM International Parallel and Distributed Processing Symposium (IPDPS),* 2010.

[11] B.A. Bauer, J.E. Davis, M. Taufer, and S. Patel. Molecular dynamics simulations of aqueous Ions at the liquid-vapor interface accelerated using graphics processors. *J. Comput. Chem.* **32**(3): 375 – 385, 2011.

[12] Y. He and C.H.Q. Ding. Using accurate arithmetics to improve numerical reproducibility and stability in parallel applications. In *Proceedings of the 14th International Conference on Supercomputing (ICS)*, 2000.

[13] M. Braxenthaler, R. Ron Unger, D. Auerbach, J.A. Given, and J. Moult. Chaos in Protein Dynamics. *Proteins*, **29**(417–425), 1997.